Реализация перемотки (DVR) при проигрывании HLS

При проигрывании HLS сегменты текущего плейлиста могут кэшироваться браузером. Это дает возможность организовать обратную перемотку на определенное время (функционал простого DVR) в HLS плеере. Рассмотрим реализацию DVR на примере VideoJS.

При создании VideoJS плеера необходимо указать следующие параметры

code

```
const LIVE_THRESHOLD = 5;
const LIVE_TOLERANCE = 5;
...
const initVideoJsPlayer = function(video) {
    let videoJsPlayer = videojs(video, {
        ...
        liveui: true,
        liveTracker: {
            trackingThreshold: LIVE_THRESHOLD,
            liveTolerance: LIVE_TOLERANCE
        },
        ...
    });
    ...
    return videoJsPlayer;
}
```

Здесь:

- `liveui: true` - включает интерфейс для перемотки
- `liveTracker.trackingThreshold` - настраивает минимальное время в секундах, котороее плеер должен проиграть, прежде чем покажет интерфейс перемотки
- `liveTracker.liveTolenrace` - настраивает время в секундах, в течение которого считается, что плеер играет живую трансляцию

лубина обратной перемотки зависит от количества сегментов в плейлисте и размера сегмента плейлиста. Эти параметры настраиваются на стороне сервера

```
hls_list_size=8
hls_time_min=2000
```

По умолчанию, максимальное время, на которое можно перемотать назад, составляет 16 секунд

```
8 * 2000 = 16000
```

Чтобы увеличить глубину обратной перемотки, необходимо увеличить размер плейлиста

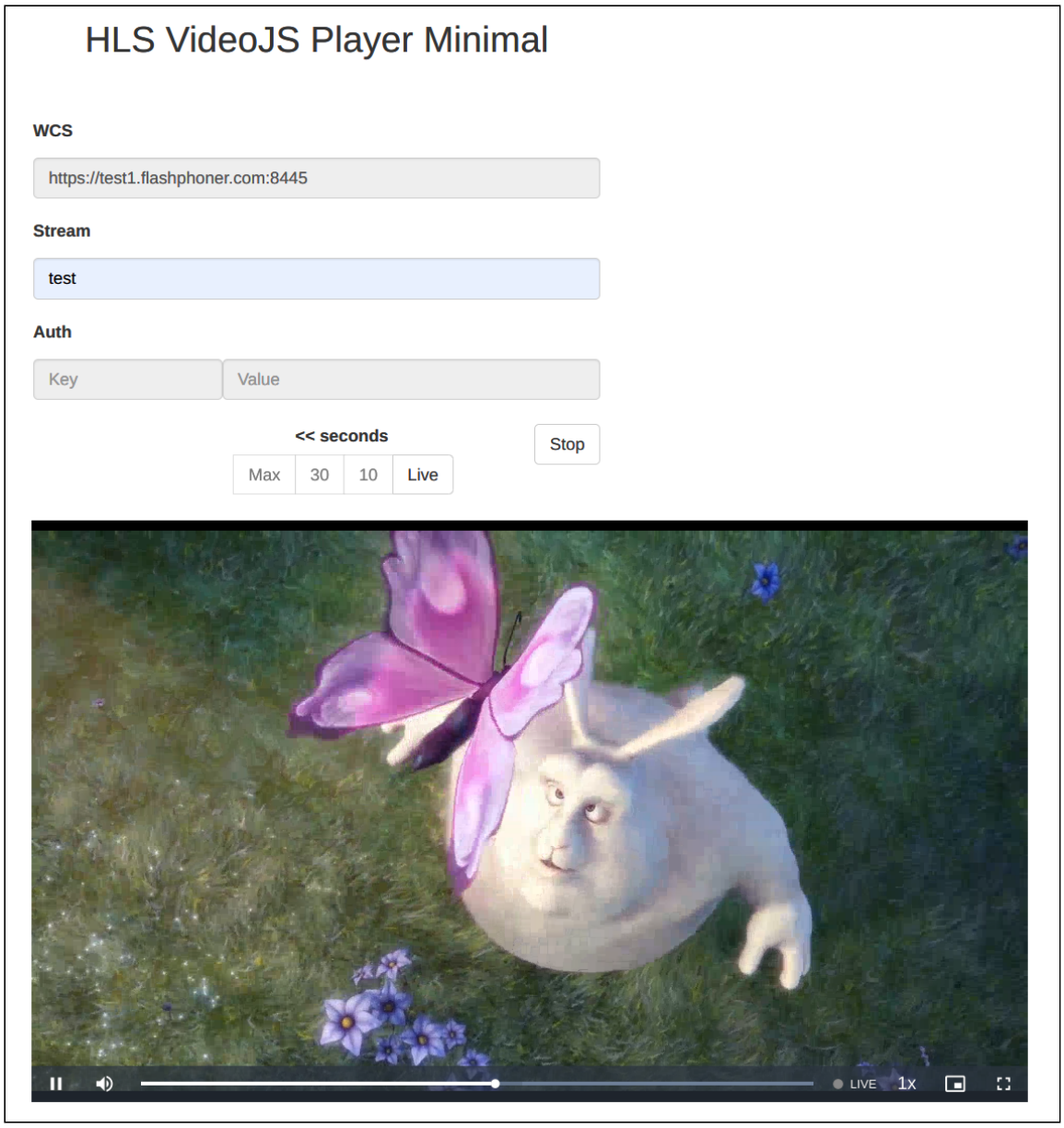
```
hls_list_size=30
```

Размер сегмента изменять не рекомендуется, поскольку в некоторых браузерах (Safari) при других значениях поток может перестать играть.

Для того, чтобы перемотать текущий поток, используется метод `Player.currentTime()`. Для получения максимально возможного диапазона перемотки используется метод `Player.seekable()`

code

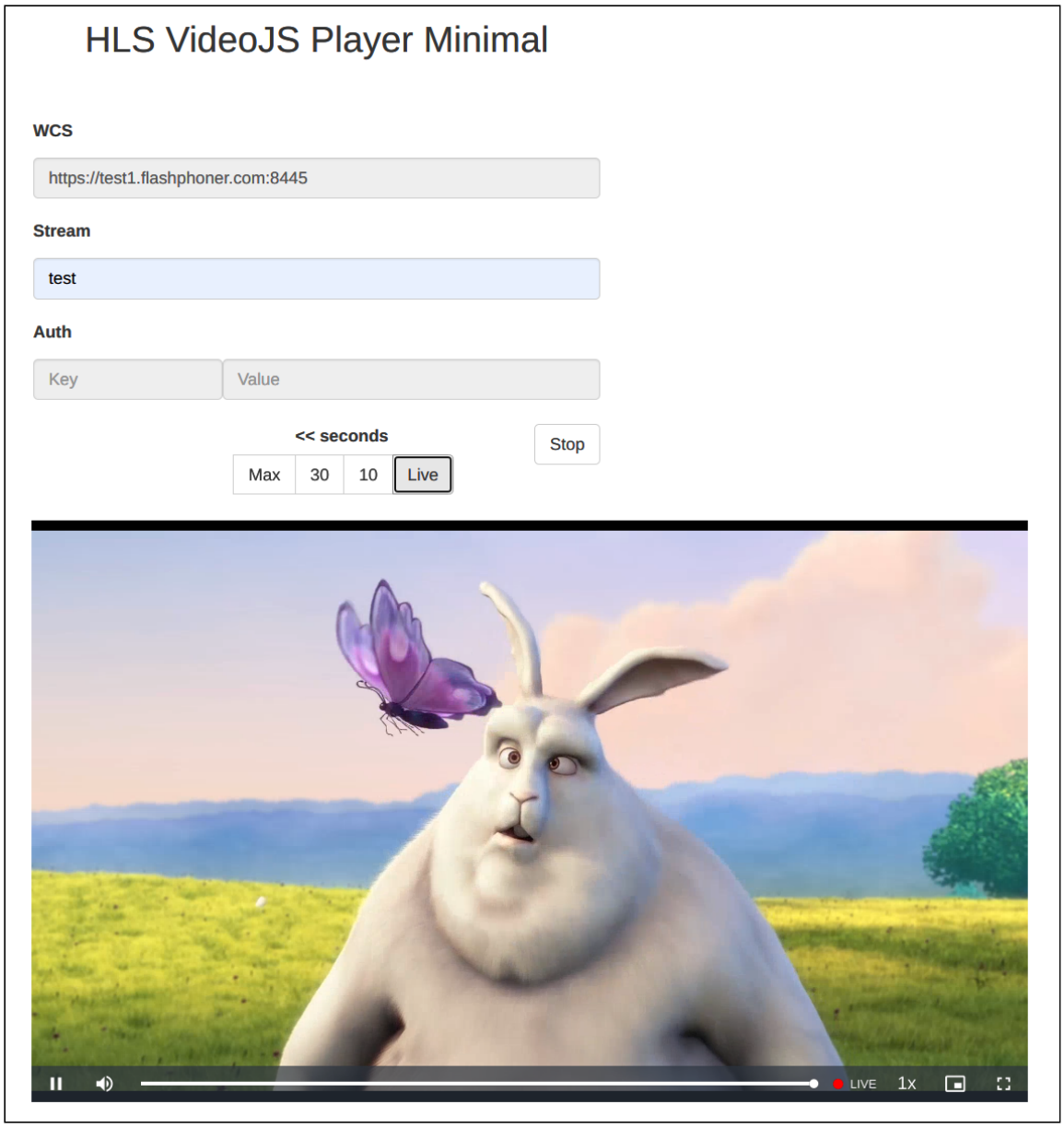
```
const backBtnClick = function(event) {
    if (player != null && player.liveTracker) {
        ...
        let seekable = player.seekable();
        let backTime = -1;
        if (event.target.id.indexOf("10") !== -1) {
            backTime = player.currentTime() - 10;
        } else if (event.target.id.indexOf("30") !== -1) {
            backTime = player.currentTime() - 30;
        }
        if (backTime < 0) {
            backTime = seekable ? seekable.start(0) : player.currentTime();
        }
        player.currentTime(backTime);
    }
}
```



Чтобы вернуться к проигрыванию живого потока, используется метод `Player.liveTracker.seekToLiveEdge()`

code

```
const liveBtnClick = function() {
    if (player != null && player.liveTracker) {
        player.liveTracker.seekToLiveEdge();
        ...
    }
}
```



Для первого подписчика на HLS поток интерфейс перемотки может не включаться автоматически, поэтому рекомендуется принудительно вызвать метод `Player.liveTracker.seekToLiveEdge()` через определенное время после начала проигрывания

code

```
const liveUIDisplay = function() {
    stopLiveUITimer()
    if (player && player.liveTracker) {
        liveUITimer = setInterval(function() {
            if (!player.liveTracker.isLive() &&
player.liveTracker.liveWindow() > LIVE_THRESHOLD) {
                // Live UI is not displayed yet, seek to live edge to display
                player.liveTracker.seekToLiveEdge();
            }
        }, LIVE_UI_INTERVAL)
```

```
    }
}
```